**Členové týmu**

* Vít Jánoš (zodpovědný za PWM)
* Vojtěch Kudela (zodpovědný za obsluhu a správu GitHub)
* Jakub Kupčík (zodpovědný za získání dat)
* Antonín Putala (zodpovědný za GUI a UART)

**Teoretický popis a vysvětlení**

V době prudkého nárůstu světové populace jsou hledány způsoby, jak zvýšit světovou produkci potravin. Jednou z možností, která se nabízí je pěstování ve **sklenících**. Velkou výhodou skleníků je, že potravinová produkce v nich není vázána na vegetační cyklus ani na klimatické podmínky. Podmínky uvnitř skleníku je možné udržovat systémem senzorů a akčních členů, které zajistí **optimální klima** pro zde pěstované rostliny.

Jak známo rostliny ke svému životu potřebují [[10]](https://www.zjistujeme.cz/co-potrebuji-rostliny-k-zivotu-jake-jsou-podminky-pro-jejich-zivot/#:~:text=Co%20pot%C5%99ebuj%C3%AD%20rostliny%20k%20%C5%BEivotu%20%E2%80%93%20Jak%C3%A9%20jsou,slune%C4%8Dn%C3%ADho%20sv%C4%9Btla.%20...%204%20Prostor%20a%20%C4%8Das%20):

1. vodu a živiny,
2. vzduch a půdu,
3. světlo a teplo,
4. prostor a čas.

Půdu a živiny je nutné zajistit při setí nebo sadbě. Vzduch bude zajištěn přístupem čerstvého vzduchu. Požadavek na prostor je omezující, co se týče rostlin, které jsme schopni na ploše vysadit a čas bohužel regulovat nelze, je tedy nezbytné nechat rostlinu růst do doby, než ponese plody.

Naopak je možné regulovat **teplotu** okolí, **světlo** a vodu, která se projeví jako **půdní vlhkost**. Tento projekt se zaměřuje na pěstování **tropických rostlin**. Je potřeba myslet na to, že tropické rostliny rostou ve velmi vlhkém prostředí, proto je nutné regulovat také **vlhkost vzduchu** [[11]](http://aaapocasi.cz/podnebne-klimaticke-pasy/).

Pro otestování možností bylo realizováno zařízení schopné, jak **měřit**, tak i **regulovat** zvolené veličiny v skleníku. Tyto veličiny je pochopitelně nutné v čase měnit, aby pokud možno odrážely, denní cykly a co nejlépe odrážely klima, ve kterém rostlina přirozeně roste. Protože jako pěstitel nemáme možnost neustále sledovat vývoj veličin, naměřené **hodnoty veličin** je vhodné **ukládat** pro další zpracování. Celé zařízení bylo naprogramováno v rozhraní [Platform.io](https://platformio.org/?utm_source=platformio&utm_medium=piohome) pro mikrokontroler [ATMEGA328P](https://www.microchip.com/en-us/product/ATmega328p). Debugování a testování bylo provedeno na desce [Arduino UNO](https://www.arduino.cc/).

Byla snaha, aby obsluha zařízení byla uživatelsky přívětivá. Proto namísto použití tlačítek, nebo maticové klávesnice bylo zvoleno **ovládání prostřednictvím osobního počítače**, k čemuž slouží grafické uživatelské rozhraní (GUI). Mezi velké výhody tohoto řešení patří velká variabilita funkcí a snadná adaptovatelnost pro budoucí modifikace. Toto rozhraní mimo jiné umožňuje zpětně zobrazit naměřené hodnoty pozorovaných veličin.

**Hardwarový popis**

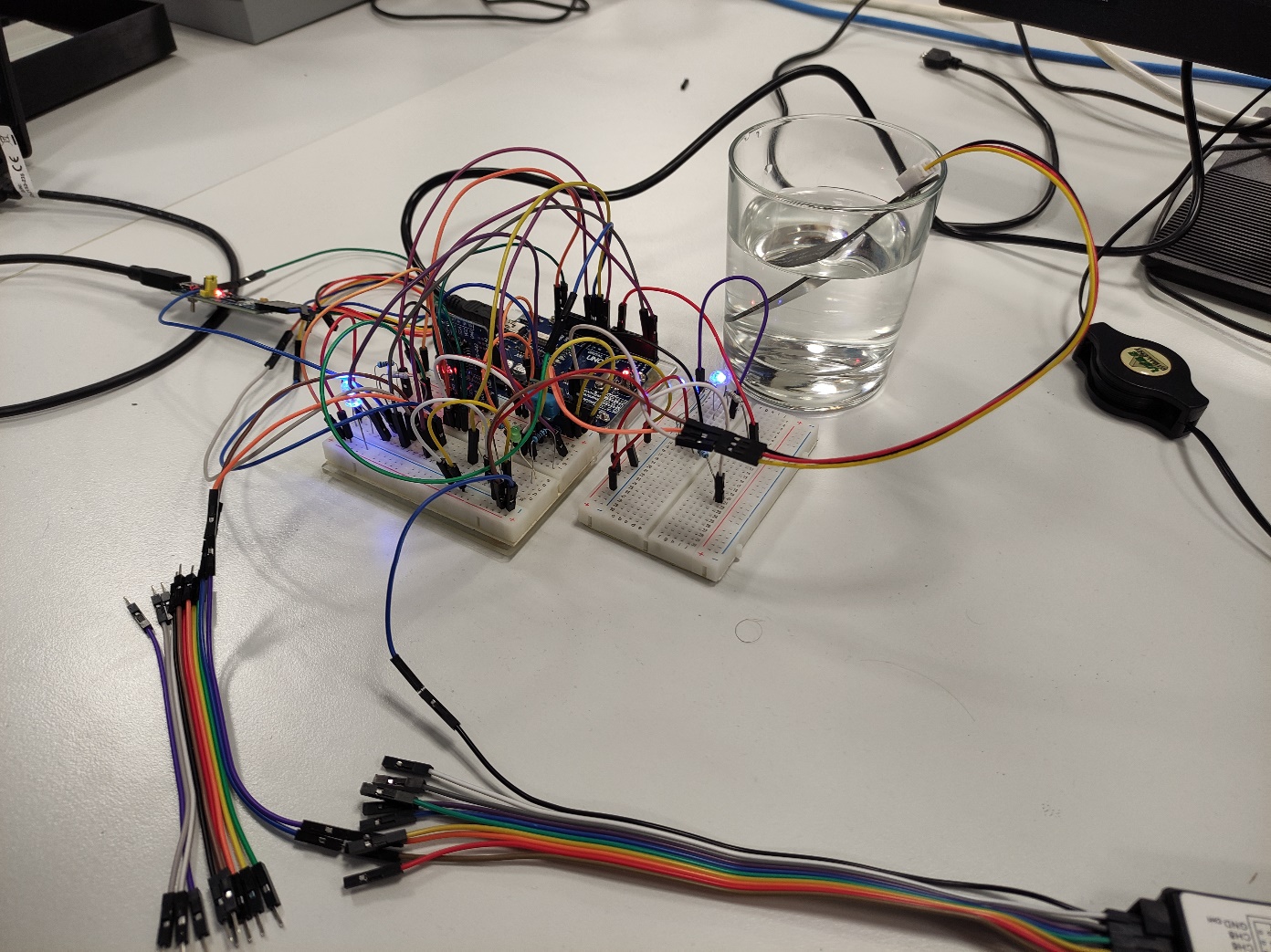
Zařízení představuje prototyp měřicího a řídicího členu pro tropický skleník. Umožňuje měření teploty, osvětlení, půdní vlhkosti a vlhkosti vzduchu. Tyto hodnoty jsou odečítány v reálném čase. Odečítání hodnot veličin zajišťují tyto senzory:

1. [DHT12](https://www.arduinolearning.com/code/dht12-temperature-sensor-arduino-example.php) – kombinované teplotní a vlhkostní čidlo,
2. [Kapacitní senzor půdní vlhkosti v1.2](https://projecthub.arduino.cc/lucasfernando/soil-moisture-sensor-comple-guide-b9c82b) – Arduino,
3. Fotorezistor.

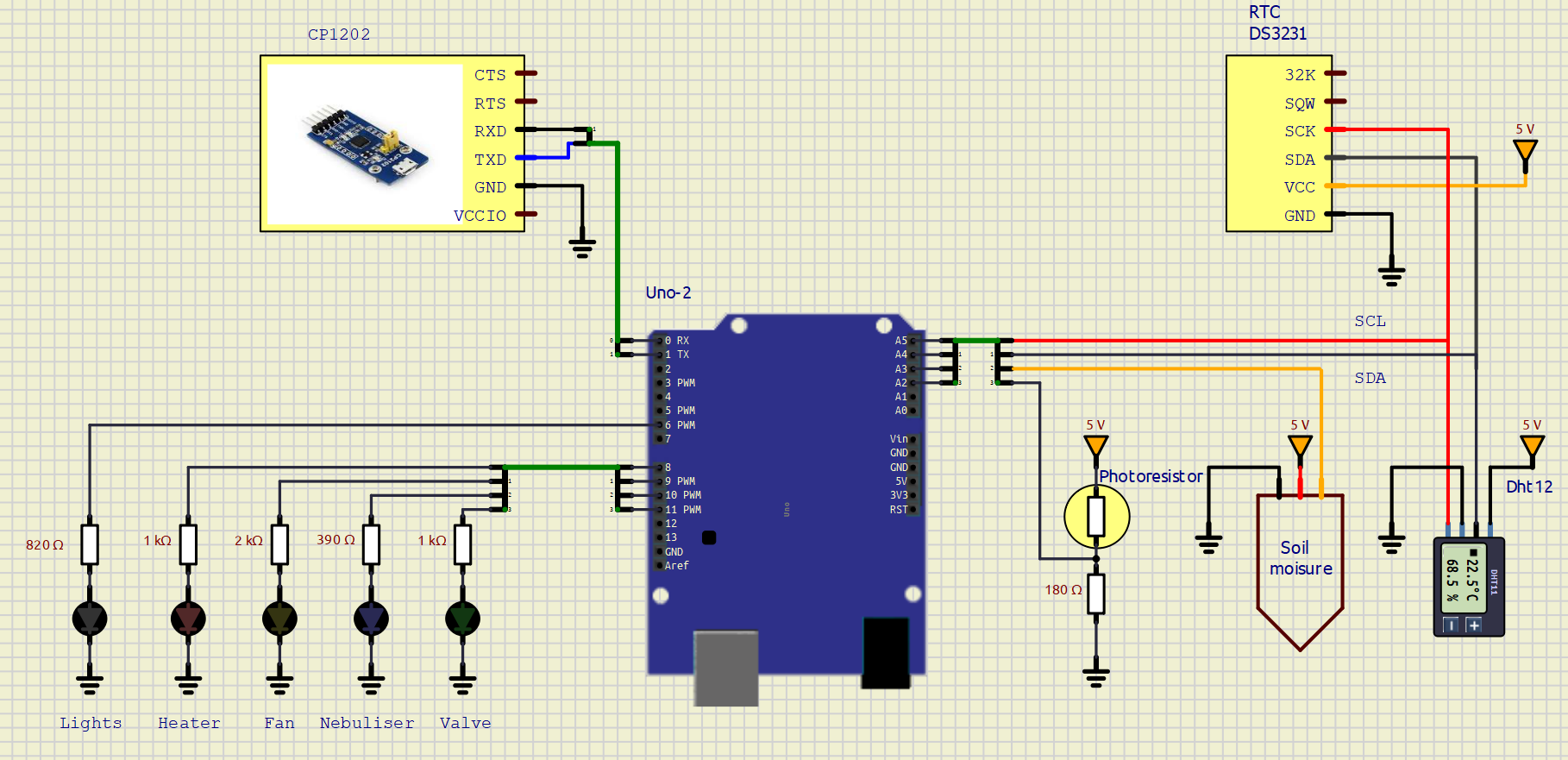
Aby odečítané hodnoty byly v reálném čase, byly použity hodiny reálného času RTC [DS3213](https://www.analog.com/media/en/technical-documentation/data-sheets/DS3231.pdf).

Další funkcí je regulace těchto veličin. K tomu slouží výstupní periférie. V tom to případě byl uvažován **topný člen**, **ventilátor**, **nebulizér**, **ventil** pro ovládání hadice, která bude zajišťovat závlahu a LED pásek, který bude představovat osvětlení. Regulaci teploty zajišťuje topný člen a ventilátor a regulaci vlhkosti vzduchu zajišťuje nebulizér a ventilátor.Pro otestování funkce byly tyto periferie nahrazeny různobarevný **LED** diodami.

**Schéma + fotka testovacího zařízení**



**Obr. 1 Prototyp měřicího a regulačního zařízení pro skleník**



**Obr. 2 Schéma zapojení**

**Periferie**

Následující odstavce popisují použité periférie a jejich zapojení.

### Kombinované teplotní a vlhkostní čidlo

***Obsah obrázku text, elektronika, Elektronické inženýrství, Elektronická součástka

Popis byl vytvořen automaticky***

**Obr. 3 Kombinované teplotní a vlhkostní čidlo**

Pro měření teploty i vlhkosti vzduchu bylo použito kombinované čidlo [DHT12](https://www.arduinolearning.com/code/dht12-temperature-sensor-arduino-example.php). Velkou výhodou tohoto čidla je zabudovaný AD převodník a komunikace pomocí I2C. Toto čidlo tedy zabere na desce pouze dva piny a je možné připojit množství dalších periférii, aniž by rostly požadavky na počet pinů desky. Vodiče I2C je nutné připojit k napájecímu napětí přes pull-up rezistory. Jsou použity vnitřní pull-up rezistory mikrokontroleru.

Pro správnou funkci je nezbytné připojit pin SCL na port PC5 a pin SDA na port PC4. Čidlo je také nutné napájet 5 V. Z tohoto čidla je pouze čteno. Zápis probíhá pouze za účelem nastavení místa paměti, ze kterého bude čteno. Toto čidlo, podobně jak je to běžné u jiných zařízení typu Slave, je že čítač paměti je automaticky inkrementován po každém čtení. Je tedy možné během jediného čtení přečíst veškerý obsah paměti čidla.

**Tab. 1 Paměť čidla** [**DHT12**](https://www.arduinolearning.com/code/dht12-temperature-sensor-arduino-example.php)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Adresa | 0x00 | 0x01 | 0x02 | 0x03 | 0x04 |
| Obsah  paměti | Vlhkost – celá část | Vlhkost – desetinná část | Teplota – celá část | Teplota – desetinná část | Kumulativní součet |

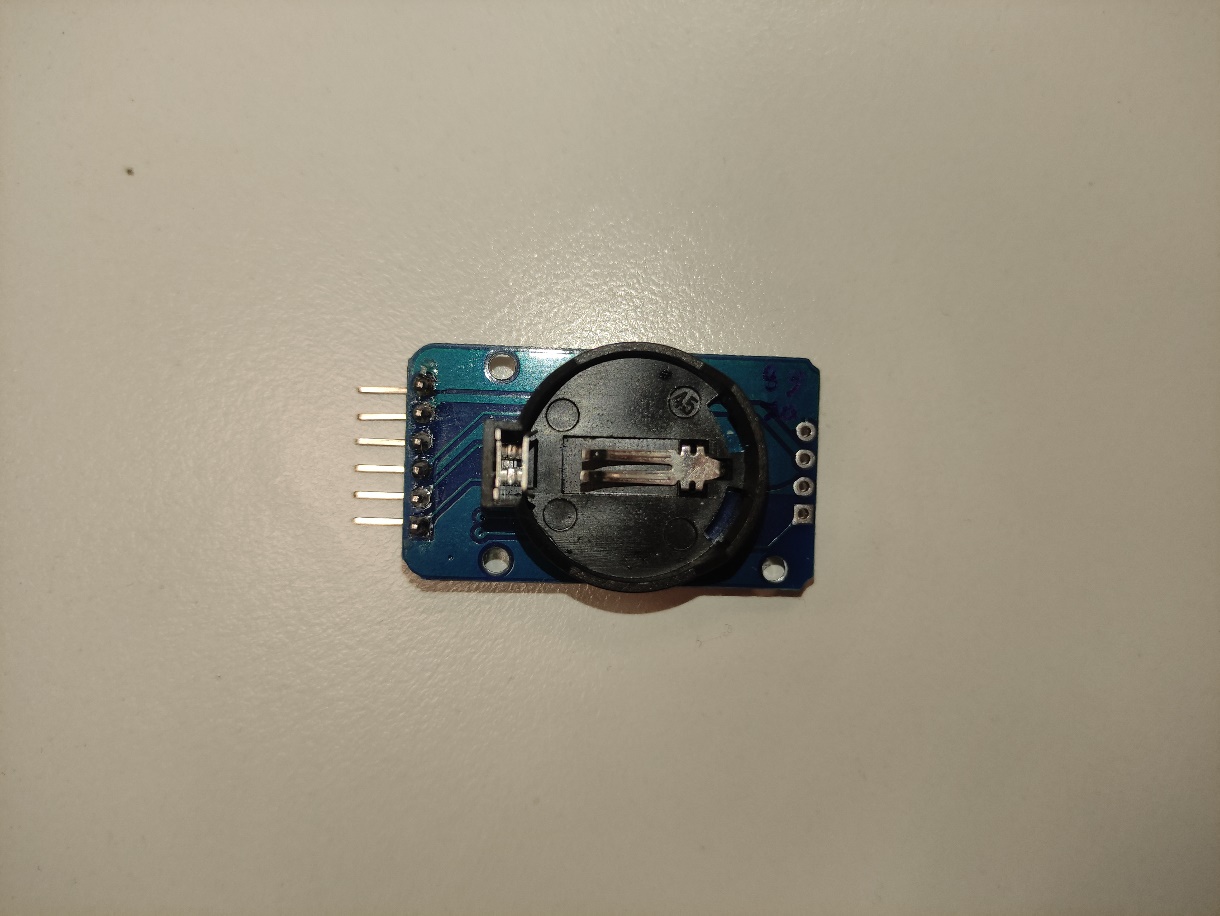
Protože operace s desetinnými čísly, především s plovoucí desetinou čárkou, jsou pro mikrokontroler velmi náročné, jsou změřené hodnoty v paměti uloženy zvlášť desetinná, zvlášť celá část. Kumulativní součet slouží ke kontrole přenesených dat. Platí:

.

Jednotlivé veličiny jsou vypsány dle pořadí v tabulce.

Neméně důležitou informací je I2C adresa čidla, která umožňuje mikrokontroleru (Masteru) komunikovat přímo s tímto čidle. Čidlo má adresu 0x5c.

### Hodiny reálného času



**Obr. 4 Hodiny reálného času**

Hodiny reálného času řeší problém časování. Naměřeným hodnotám veličin je přiřazen čas. Data jsou vysílána přibližně každou sekundu. Ovšem program je napsán v jazyce C nikoli v jazyku symbolických adres a zařízení není optimalizováno, aby přesně každou sekundu poslalo zprávu. Z tohoto důvodu není možné tento signál použít k synchronizaci dat. Další nespornou výhodou je, že hodiny reálného času podávají informaci také o dnu, měsíci a roku, což velmi zjednodušuje následné zpracování údaje o času.

Použité hodiny reálného času [DS3213](https://www.analog.com/media/en/technical-documentation/data-sheets/DS3231.pdf) umožňují stejně jako teplotní a vlhkostní čidlo komunikovat přes I2C. Připojení je obdobné jako v případě teplotně-vlhkostního čidla. V tomto případě je nutné, jak data z hodin číst, tak i zapisovat, aby bylo možné nastavit aktuální čas.

Pro čtení i zapisování je nutné znát strukturu paměti, která je dána v následující tabulce:

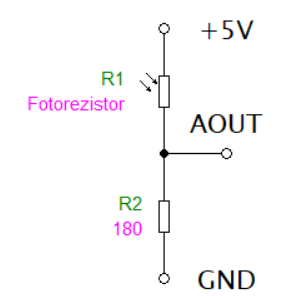
**Tab. 2 Struktura paměti** [**DS3213**](https://www.analog.com/media/en/technical-documentation/data-sheets/DS3231.pdf)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Adresa | Bit | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0x00 | 0 | Desítky sekund | | | Sekundy | | | |
| 0x01 | 0 | Desítky minut | | | Minuty | | | |
| 0x02 | 0 | 0 | Desítky hodin | | Hodiny | | | |
| 0x03 | 0 | 0 | 0 | 0 | 0 | Den v týdnu | | |
| 0x04 | 0 | 0 | Desítky dnů v týdnu | | Den v měsíci | | | |
| 0x05 | 1 | 0 | 0 | Desítky měsíců | Měsíce | | | |
| 0x06 | Desítky let | | | | Roky | | | |

Takto nastavené RTC bude čítat v 24 hodinovém formátu. Je nastaveno 21.století. I2C adresa hodin je 0x68.

### Čidlo osvětlení

Čidlo osvětlení je realizováno pomocí děliče, přičemž odpor, na kterém není měřeno napětí je nahrazen fotorezistorem. Tato konfigurace byla zvolena, aby s rostoucím osvětlením rostlo napětí na výstupu AOUT. Protože je výstup analogový bylo toto čidlo připojeno k analogovému vstupu. Konkrétně byl zvolen PC2.



**Obr. 6 Schéma zapojení čidla osvětlení**

**Obr. 5 Fotorezistor**

Tato konfigurace není sama o sobě schopna dát smysluplnou hodnotu osvětlení, je nutný přepočet. Byla změřena závislost výstupního napětí vyjádřeného 10-bitový číslem na osvětlení. Měření probíhalo pouze pro tmu a maximální osvětlení, což bylo provedeno tak, že dioda svítila na plný výkon do fotorezistoru. Tyto dva body byly proloženy přímkou. Samozřejmě se jedná o aproximaci, ve skutečnosti je závislost nelineární. Nicméně pro orientační zjištění hodnoty osvětlení a přibližné nastavení spínacího prahu je taková aproximace zcela dostačující.

**Tab. 3 Hodnoty pro aproximaci převodní charakteristiky čidla osvětlení**

|  |  |  |
| --- | --- | --- |
| *N* [‒] | 164 | 480 |
| *E* [lx] | 16 | 358 |

Převodní charakteristika byla aproximována následující funkcí:



**Obr. 7 Závislost zobrazené hodnoty osvětlení na výstupní hodnotě ADC**

Zvolená maximální hodnota osvětlení působí poměrně zvláštně. Nicméně tato hodnota byla zvolena pouze proto, aby nebylo nutné řešit operace s desetinnými čísly, neboť tyto operace jsou pro mikrokontroler podstatně náročnější.

### Čidlo půdní vlhkosti

Obsah obrázku text, elektronika, Elektronické inženýrství, kabel

Popis byl vytvořen automaticky

**Obr. 8 Čidlo půdní vlhkosti**

Měření půdní vlhkosti je realizováno pomocí [kapacitního čidla půdní vlhkosti](https://projecthub.arduino.cc/lucasfernando/soil-moisture-sensor-comple-guide-b9c82b) z Arduina. Toto čidlo má napájení 5 V, zemi a analogový výstup. Ten byl připojen na analogový pin PC3. K tomuto čidlu bohužel není k dostání převodní charakteristika, a tak byla určena experimentálně a aproximována přímkou. Jako minimální vlhkost byl uvažován stav, když bylo čidlo na vzduchu. Maximální vlhkost byla uvažována, pokud bylo čidlo umístěno ve sklenici vody.

**Tab. 4** **Hodnoty pro aproximaci převodní charakteristiky čidla půdní vlhkosti**

|  |  |  |
| --- | --- | --- |
| *N* [‒] | 233 | 186 |
| *SM* [%] | 0 | 100 |

Převodní charakteristika byla aproximována následující funkcí:



**Obr. 9 Závislost zobrazené hodnoty půdní vlhkosti na výstupní hodnotě ADC**

Stanovený přepočet je opět pouze orientační. V žádném případě se nejedná o přesnou fyzikální hodnotu.

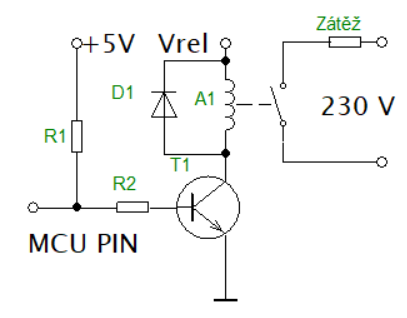
### Topná člen, ventilátor, nebulizér a ventil

Při regulaci obou vlhkostí a teploty lze s výhodou využít setrvačnosti prostředí, samotný prostor skleníku bude udržovat teplotu i vlhkost. Z toho důvodu přesná PWM regulace nemá význam, postačujícím řešením je **on/off regulace**. Jednotlivé periférie budou připojeny na následující piny:

* PB0 ‒ topná člen,
* PB1 ‒ ventilátor,
* PB2 ‒ nebulizér,
* PB3 ‒ ventil pro zavlažování.

Protože použité periférie vyžadují poměrně vysoký spínací výkon je vhodné použít následující konfiguraci. Mikrokontroler spíná tranzistor a ten spíná relé, které umožňuje spínat velké výkony. Pro tuto aplikaci možné využít bipolární tranzistor. Je nutné uvažovat, jak velký proud bude tranzistor spínat, aby bylo možné správně tranzistor výkonově dimenzovat. Pokud by byl použit NMOS, byl by vhodný driver, protože tyto tranzistory vyžadují vyšší hodnotu spínacího napětí, než je schopen mikrokontroler poskytnout. Tranzistor je nutné budit dostatečným proudem, proto je na bázi připojen pull-up rezistor. V některých případech lze použít vnitřní pull-up mikrokontroleru. Dále je nutné omezit proud do báze bázovým rezistorem.

Relé je pro tuto aplikaci dostatečně rychlé, problém ale může způsobit, že po určitém počtu sepnutí relé odejde, změna může v nejhorším případě nastat každou sekundu. Abychom bylo této situaci zabráněno, je vždy nastavena hystereze regulované veličiny. Pokud by velký počet spínacích cyklů byl prioritou, stálo by za to zvážit použití optočlenu, nebo SSR relé. Nepříjemnou vlastností relé, je velký záporný napěťový překmit po rozepnutí. Tento překmit způsobí průraz spínacího tranzistoru. Řešením je například **dioda** zapojená paralelně s cívkou relé.



**Obr. 10 Doporučené připojení výkonových zařízení**

Toto zapojení je **aktivní ve vysoké úrovni**. Pro testování tyto periférie nebyly k dispozici, byly modelovány pomocí diod, které byly tedy též zapojeny, aby byly aktivní ve vysoké úrovni.

Byly použity tyto barvy:

1. Červená – topná člen
2. Oranžová – ventilátor
3. Zelená – ventil
4. Modrá – nebulizér

### LED pásek

Poněkud těžší úkol představuje regulace osvětlení. Pokud chceme nastavit osvětlení, je nutné nastavit přímo regulovat intenzitu světelného zdroje, protože v tomto případě se neuplatní setrvačnost prostředí. V tomto případě se nabízí použití **PWM regulace**, protože tuto funkci mikrokontroler umožňuje.

LED pásek je nutné připojit k portu PD6. Bylo uvažováno zapojení **active-high**. Je potřeba zvážit odběr LED pásku. Napájení kratších rozměrů zajistí port mikrokontroleru, pro větší odběru je vhodné použít spínací tranzistor. Pro účely testování byl LED pásek nahrazen modrou LED diodou.

**Komentář: obsluhu zajišťuje PC**

Komunikaci s PC je realizována prostřednictvím sériového portu. Převod z USB na UART zajišťuje převodník [CP2102](https://www.alldatasheet.com/html-pdf/201067/SILABS/CP2102/215/1/CP2102.html).

Obsah obrázku text, Obvodoví součástka, Elektronická součástka, Pasivní součástka

Popis byl vytvořen automaticky

**Obr. 11 Převodník** [**CP2102**](https://www.alldatasheet.com/html-pdf/201067/SILABS/CP2102/215/1/CP2102.html)

Počítač je na převodník připojen prostřednictvím microUSB. Pro zajištění komunikace s deskou je nutné propiji země převodníku a desky. Dále musí být propojeny datové piny. Vysílací pin TXD převodníku bude připojen na přijímací pin desky RX, zatímco přijímací pin RXD bude připojen na vysílací pin desky TX. Na převodníku byl jumper umístěn tak, aby výstupní napětí na UARTu bylo 5 V.

Pro komunikaci s deskou je nutné používat aplikaci [Tropical plants](https://github.com/VojtaKudela/BPC-DE2/tree/main/Python_GUI), která obsahuje grafické rozhraní uzpůsobené řízení skleníku.

**Softwarový popis**

Pro funkci zařízení bylo nezbytné vytvořit jak program pro mikrokontroler, tak pro osobní počítač. Zatímco program pro mikrokontroler byl napsán v jazyce C, program pro osobní počítač byl napsán v Pythonu.

**Python**

Úkolem programu pro osobní počítač je sbírat data vysílaná zařízením a vysílat instrukce k regulaci měřených veličin. Aplikace též umožňuje změnit čas zařízení a graficky zobrazit změřené hodnoty. Samotnou aplikaci spustí soubor [Tropical\_plants](https://github.com/VojtaKudela/BPC-DE2/blob/main/Python_GUI/Tropical_plants.py). Tento program vytvoří aplikaci [GUI\_main](https://github.com/VojtaKudela/BPC-DE2/blob/main/Python_GUI/GUI_main.py) a spustí ji.

Podrobný popis všech použitých tříd a funkcí se nachází v této [dokumentaci](https://raw.githack.com/VojtaKudela/BPC-DE2/refs/heads/main/Documentation/Python/html/index.html).

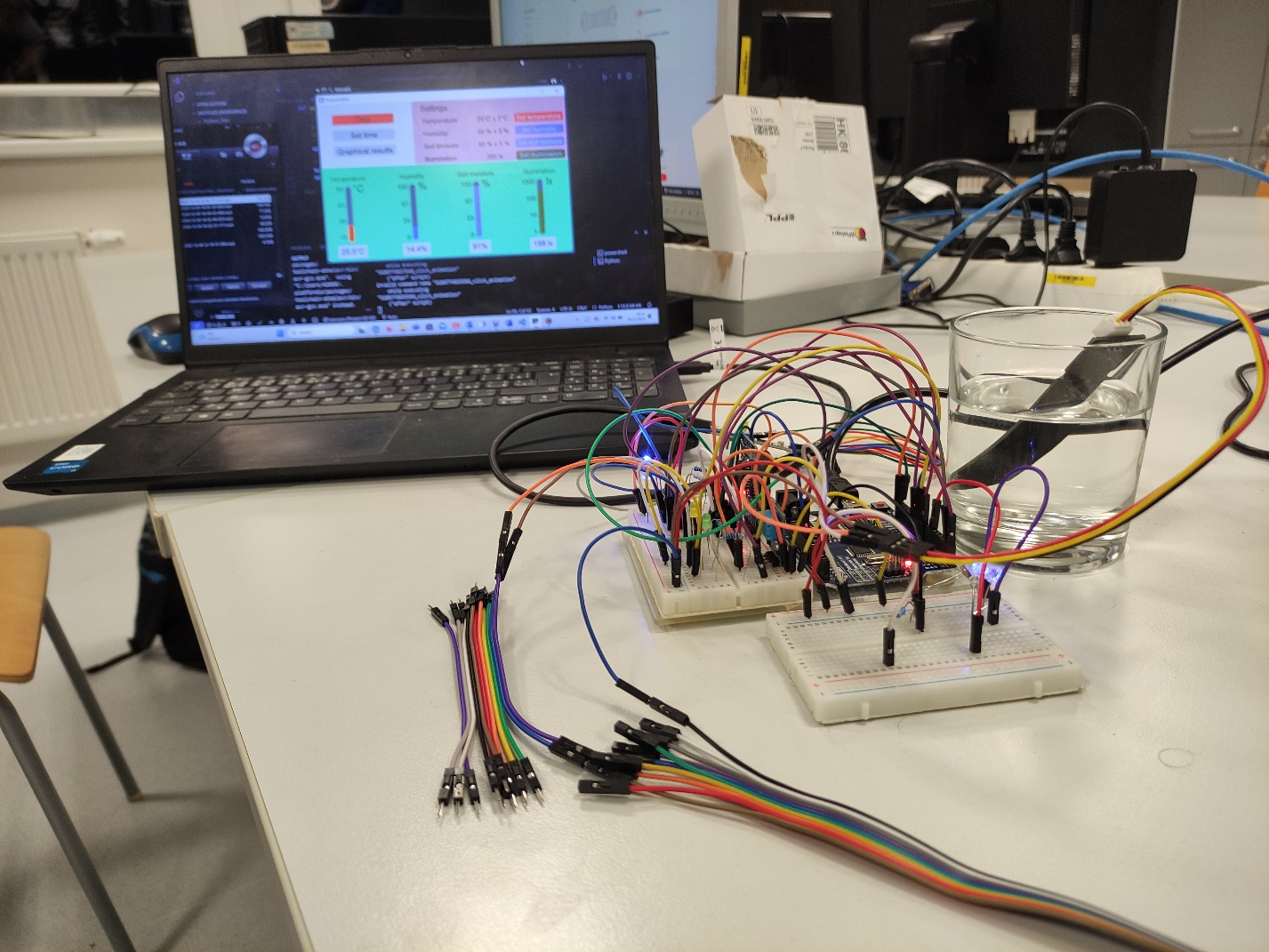
### [GUI\_main](https://github.com/VojtaKudela/BPC-DE2/blob/main/Python_GUI/GUI_main.py)

Jedná se o třídu, která obsahuje funkce pro vytvoření GUI, obsluhu sériové komunikace a tvorbu dalších vyskakovacích oken.

Při inicializaci třídy dojde k vytvoření GUI. K tomu slouží funkce [GUI\_create](https://github.com/VojtaKudela/BPC-DE2/blob/main/Python_GUI/GUI_create.py), která volá funkce [createGUImain](https://github.com/VojtaKudela/BPC-DE2/blob/main/Python_GUI/createGUImain.py), [createGUIset](https://github.com/VojtaKudela/BPC-DE2/blob/main/Python_GUI/createGUIset.py) a [createGUIshow](https://github.com/VojtaKudela/BPC-DE2/blob/main/Python_GUI/createGUIshow.py). Funkce [createGUImain](https://github.com/VojtaKudela/BPC-DE2/blob/main/Python_GUI/createGUImain.py) vytváří tlačítka pro připojení, zobrazení grafických výsledků a nastavení času. Funkce [createGUIset](https://github.com/VojtaKudela/BPC-DE2/blob/main/Python_GUI/createGUIset.py) generuje nabídku nastavení regulovaných veličin a [createGUIshow](https://github.com/VojtaKudela/BPC-DE2/blob/main/Python_GUI/createGUIshow.py) vytvoří ukazatele naměřených hodnot.

V rozhraní se nacházejí tlačítka, při jejichž stisku se otevře nové okno:

1. Set temperature – okno vytvoří program [temp\_GUI](https://github.com/VojtaKudela/BPC-DE2/blob/main/Python_GUI/temp_GUI.py),
2. Set humidity – [hum\_GUI](https://github.com/VojtaKudela/BPC-DE2/blob/main/Python_GUI/hum_GUI.py),
3. Set soil moisure – [soil\_GUI](https://github.com/VojtaKudela/BPC-DE2/blob/main/Python_GUI/soil_GUI.py),
4. Set illumination – [ilu\_GUI](https://github.com/VojtaKudela/BPC-DE2/blob/main/Python_GUI/ilu_GUI.py),
5. Set time – [time\_GUI](https://github.com/VojtaKudela/BPC-DE2/blob/main/Python_GUI/time_GUI.py),
6. Graphical results – [graph\_GUI](https://github.com/VojtaKudela/BPC-DE2/blob/main/Python_GUI/graph_GUI.py).



**Obr. 12 Zařízení i řídicí osobní počítač**

#Ještě něco k funkcím, které běží a umožňují sériovou komunikaci#

### Sériová komunikace

Veškerá sériová komunikace probíhá na rychlost **9600 Bd**. Aby se ušetřil datový tok, jsou zasílány přímo číselné hodnoty, nikoli jejich zápis v ASCII kódu. Osobní počítač vysílá dva různé datové balíčky; první nese informaci o **nastavení času**, druhý o **nastavení** regulovaných b. Mikrokontroler vysílá pouze jeden balíček.

Balíček pro nastavení času má následující strukturu:

**Tab. 5 Datový balíček pro nastavení času**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Byte | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Obsah | 'T' | Rok | Měsíc | Den v měsíci | Den v týdnu | Hodina | Minuta | Sekunda |

První byte je ASCII kód písmene „T“. Slouží k rozpoznání. Ostatní byty musí formou odpovídat paměti RTC. Tento balíček je odeslán vždy, když uživatel nastaví v rozhraní čas. Balíček pro nastavení regulovaných veličin má tuto strukturu:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Byte | 0 | 1 | 2 | 3 | 4 |
| Obsah | 'R' | Teplota - hodnota | Teplota - hystereze | Vlhkost - hodnota | Vlhkost - hystereze |
|  |  |  |  |  |  |
| Byte |  | 5 | 6 | 7 | 8 |
| Obsah |  | Půdní vlhkost - hodnota | Půdní vlhkost - hystereze | Osvětlení - stovky | Osvětlení - jednotky |

První byte je ASCII kód písmene „R“. Slouží k rozpoznání. Ostatní byty obsahují přímo číselnou hodnotu, která bude nastavena na mikrokontroleru. **Hodnota** je úroveň, na kterou bude daná veličina regulována. **Hystereze** udává, s jakou tolerancí bude daná veličina nastavena. Pokud se bude veličina pohybovat v rozmezí hodnota ± hystereze, regulační členy nebudou aktivní. K odeslání dojde poté, co uživatel změní v rozhraní nastavenou regulovanou hodnotu libovolné veličiny.

#Datový balíček vysílaný mikrokontrolerem#

### Zpracování přijatých dat z mikrokontroleru

**Main**

**Knihovny**

**Heater**

**Soil**

**Huminity**

**PWM**

**(potřeba i nějaké obkecávačky a nějaká schémata)**

**Instrukční list**

#Fotka hlavního panelu a dalších panelů#

**Nastavení**

Pro ukázku obsluhy zařízení a jeho činnosti za chodu klikněte [zde](https://www.youtube.com/watch?v=y9z3xt5LS8A).

**Reference**

1. [Climate Chamber System](https://projecthub.arduino.cc/ms_peach/climate-chamber-system-c545de).
2. [Learning AVR-C Episode 7: PWM](https://www.youtube.com/watch?v=ZhIRRyhfhLM&list=PLA6BB228B08B03EDD&index=7).
3. [Learning AVR-C Episode 8: Analog Input](https://www.youtube.com/watch?v=51QJ_WHN7u0&list=PLA6BB228B08B03EDD&index=8&fbclid=IwY2xjawGghWBleHRuA2FlbQIxMAABHVy7dx15Emsi53adUYbmtC7HX_bKwPgDDZE106S3zNYXwdnrUu0nhW8zyA_aem_Rj_25ybcyhsOJBNBMxLZ1Q).
4. [Custom Tkinter - Official Documentation](https://customtkinter.tomschimansky.com/)
5. [pySerial's documentation](https://pyserial.readthedocs.io/en/latest/)
6. [ASCII table](https://www.ascii-code.com/)
7. [ATMEGA328P- datasheet](https://www.microchip.com/en-us/product/ATmega328p)
8. [Soil moisure](https://www.makerguides.com/capacitive-soil-moisture-sensor-with-arduino/)
9. [Arduino map()](https://reference.arduino.cc/reference/en/language/functions/math/map/)
10. [Co potřebují rostliny k životu - Jaké jsou podmínky pro jejich život | Zjišťujeme.cz](https://www.zjistujeme.cz/co-potrebuji-rostliny-k-zivotu-jake-jsou-podminky-pro-jejich-zivot/#:~:text=Co%20pot%C5%99ebuj%C3%AD%20rostliny%20k%20%C5%BEivotu%20%E2%80%93%20Jak%C3%A9%20jsou,slune%C4%8Dn%C3%ADho%20sv%C4%9Btla.%20...%204%20Prostor%20a%20%C4%8Das%20)
11. [Podnebné (klimatické) pásy - Počasí](http://aaapocasi.cz/podnebne-klimaticke-pasy/)
12. [Your Gateway to Embedded Software Development Excellence · PlatformIO](https://platformio.org/?utm_source=platformio&utm_medium=piohome)
13. [DS3213 datasheet](https://www.analog.com/media/en/technical-documentation/data-sheets/DS3231.pdf)
14. [Soil Moisture Sensor - Comple Guide | Arduino Project Hub](https://projecthub.arduino.cc/lucasfernando/soil-moisture-sensor-comple-guide-b9c82b)
15. [DHT12 temperature sensor and Arduino example - Arduino Learning](https://www.arduinolearning.com/code/dht12-temperature-sensor-arduino-example.php)
16. [Arduino - Home](https://www.arduino.cc/)
17. [CP2102 datasheet(1/18 Pages) SILABS | SINGLE-CHIP USB TO UART BRIDGE](https://www.alldatasheet.com/html-pdf/201067/SILABS/CP2102/215/1/CP2102.html)

**Seznam použitých nástrojů**

1. [VS Code](https://code.visualstudio.com/)
2. [ChatGPT](https://chatgpt.com/)
3. [Microsoft Copilot](https://copilot.microsoft.com/)
4. [SimulIDE](https://simulide.com/p/)
5. [Saleae Logic 2](https://www.saleae.com/pages/downloads)
6. [Online C compiler](https://www.online-cpp.com/online_c_compiler)
7. [Bandicam](https://www.bandicam.com/cz/)
8. [draw.io](https://app.diagrams.net/)
9. [Doxygen](https://doxygen.nl/index.html)
10. [Matlab](https://www.mathworks.com/products/matlab.html)
11. [ProfiCAD - Elektro CAD Software - ProfiCAD](https://www.proficad.cz/)